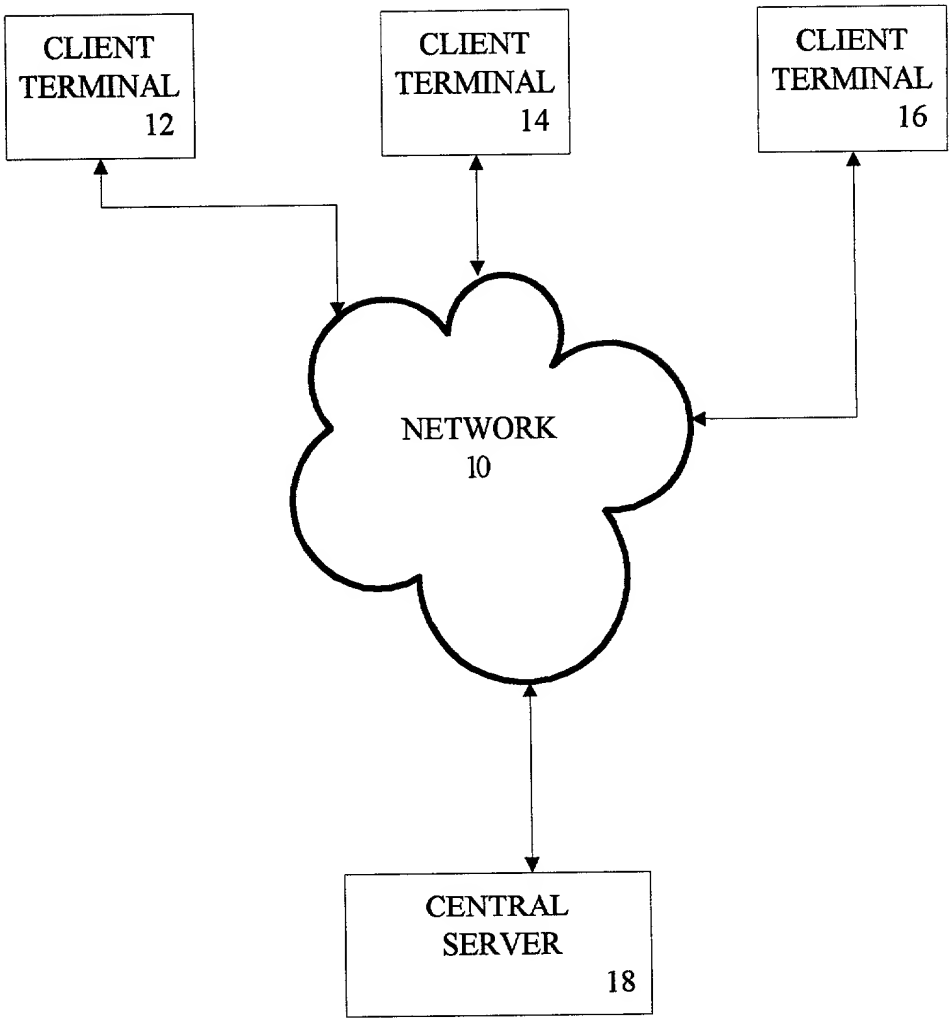
CLIENT TERMINAL 12
CLIENT TERMINAL 14
CLIENT TERMINAL 16
NETWORK 10
CENTRAL SERVER 18

FIG. 1

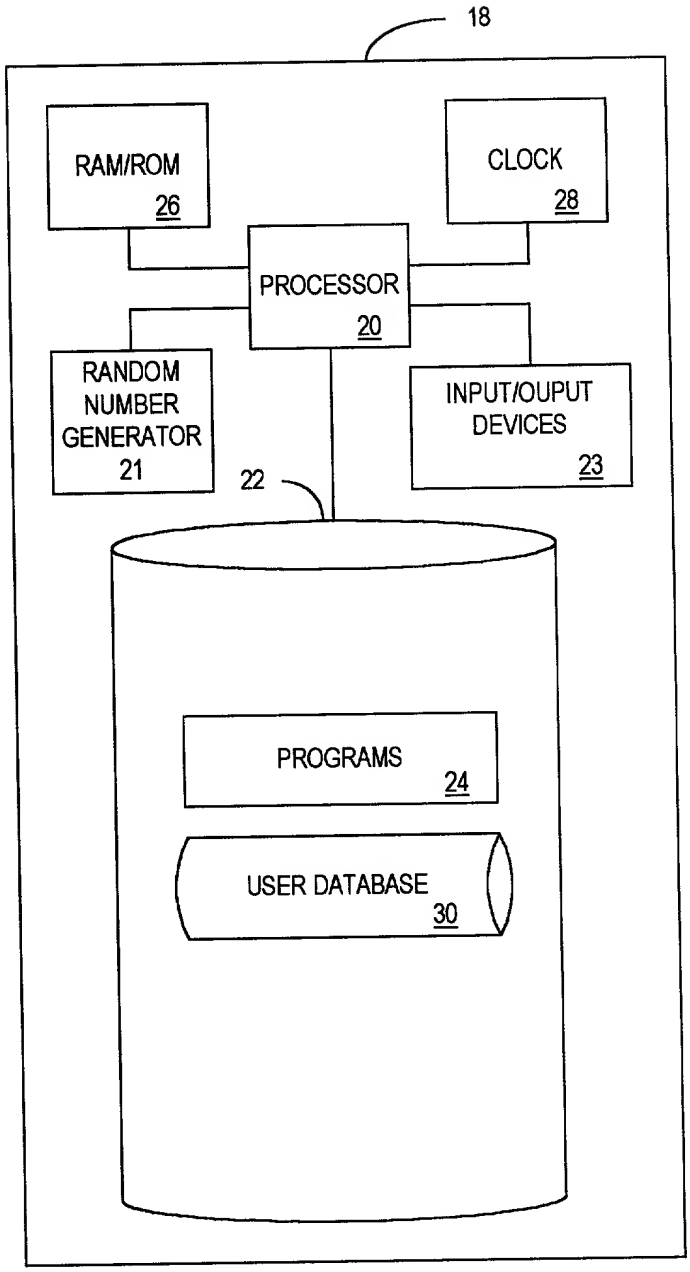
18
RAM/ROM
26
CLOCK
28
PROCESSOR
20
RANDOM
NUMBER
GENERATOR
21
INPUT/OUPUT
DEVICES
23
22
PROGRAMS
24
USER DATABASE
30

FIG. 2

USER NAME 32	USER PASSWORD 34	JLVSession COOKIE NUMBER 36	VARIABLES 38

FIG. 3

400

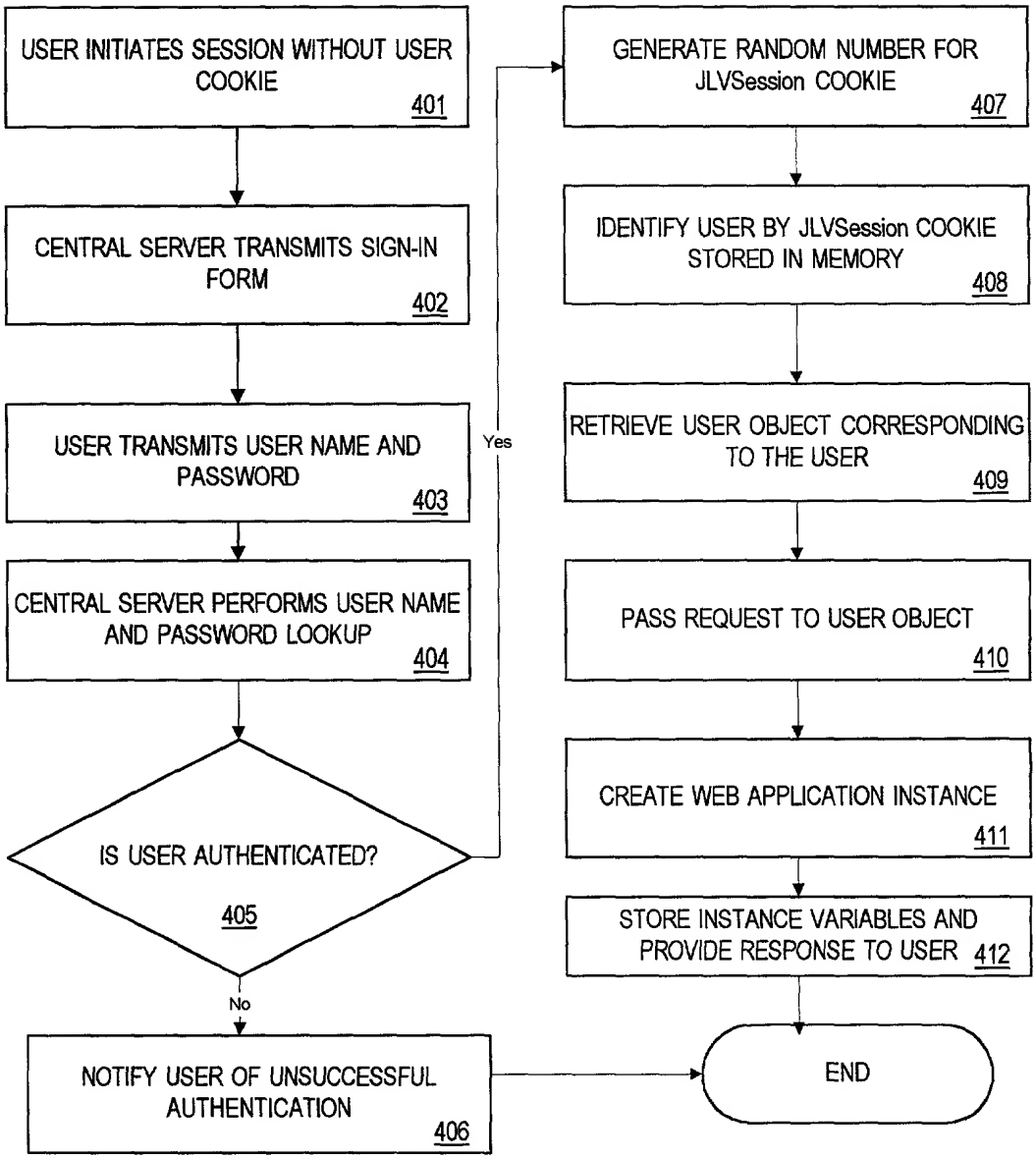

FIG. 4

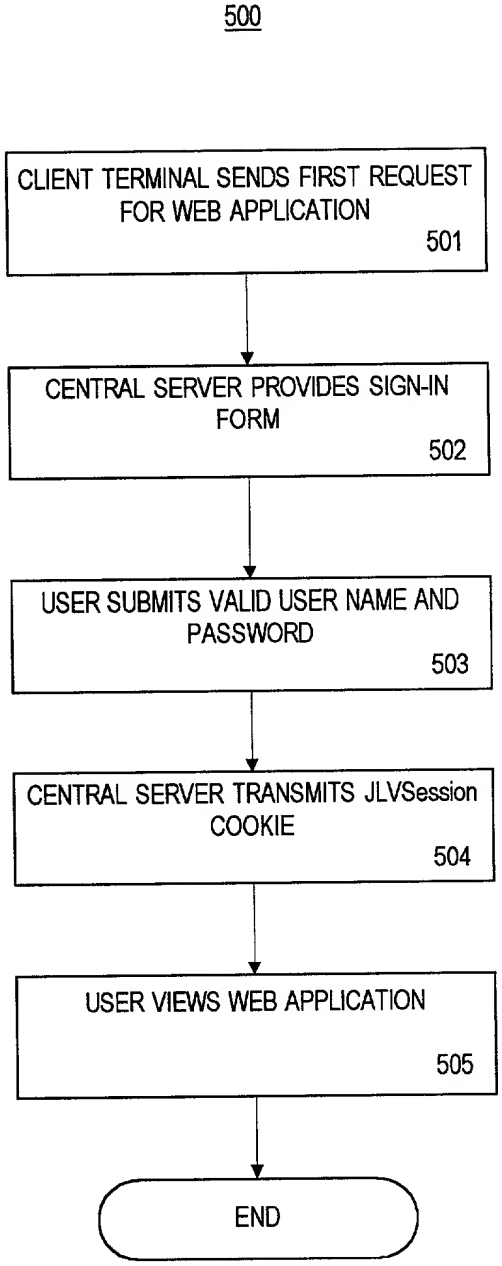

FIG. 5

```
import java.util.*;
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;

public class TraditionalDemo extends HttpServlet {
      private Vector mCookies = new Vector();
      private Hashtable mUsers = new Hashtable();
  public void service(HttpServletRequest req,HttpServletResponse res)
    throws ServletException, IOException {
            try {
                  //
                  // check for sign-in
                  //
                  String cmd = req.getParameter("Command");
                  if (cmd != null && cmd.equals("SignIn")) {
                        String username = req.getParameter("Username");
                        String password = req.getParameter("Password");
                        if (username == null) { showError(req,res,"Username not
specified."); }
                        else if (password == null) { showError(req,res,"Password
not specified."); }
                        else {
                              // this application sign-in approach has the
disadvantage
                              // that the user database and access control are
specific to
                              // this particular application and therefore not
easy to manage
                              // when there are a lot of applications
                              BufferedReader br = new BufferedReader(new
FileReader("UserDB"));
                              boolean done = false;
                              String line;
                              int i;
                              while ((line=br.readLine()) != null) {
                                    line = line.trim();
                                    if (line.equals("") || line.charAt(0) ==
'#') continue;
                                    i = line.indexOf('=');
                                    if (i == -1) continue;
                                    if (username.equals(line.substring(0,i))) {
                                          // user database without password
encryption to
                                          // simplify this demonstration
                                          if
(password.equals(line.substring(i+1))) {
                                                // authentication successful
                                                String cookie;
                                                if
(mUsers.containsKey(username)) {
                                                      // remove all cookie
                                                      for (i = mCookies.size() -
1; i >= 0; i--) {
```

(c) 2000 Hotlens.com Inc.

FIG. 6A

```
(String)mCookies.elementAt(i);                                    cookie =

(username.equals(cookie.substring(0,cookie.indexOf('.')))) {      if

      mCookies.removeElementAt(i);

                                                                        break;
                                                                  }
                                                            }
                                                      }
                                                      // send session cookie
                                                      while (true) {
                                                            cookie =
username+"."+getRandomId();

                                                            if
(!mCookies.contains(cookie)) break;

                                                      }
                                                      mCookies.addElement(cookie);
                                                      if
(!mUsers.containsKey(username)) {

                                                            Hashtable h = new
Hashtable();

                                                            // non object-oriented
initialization of user values

                                                            h.put("Balance","0");
                                                            mUsers.put(username,h);
                                                      }
                                                      Cookie c = new
Cookie("Session",cookie);

                                                      c.setPath(req.getServletPath());
                                                      res.addCookie(c);
                                                      String url =
"http://"+req.getServerName()+":"+req.getServerPort()+req.getServletPath();
                                                      String querystr =
req.getQueryString();

                                                      if (querystr != null &&
!querystr.equals("")) url += "?"+querystr;

                                                      url += ((url.indexOf('?')==-
1)?"?":"&")+"Session="+getRandomId();

                                                      // this is done so that Netscape
will not complain that response contains no data.

      res.setStatus(HttpServletResponse.SC_MOVED_TEMPORARILY);
                                                      res.setHeader("Location",url);
                                                } else {
                                                      showError(req,res,"Password not
valid.");

                                                }
                                                break;
                                          }
                                    }
                                    if (line == null) { showError(req,res,"Username
not valid."); }

                                    br.close();
                              }
                              return;
                        }
```

(c) 2000 Hotlens.com Inc.

FIG. 6B

```
                    //
                    // get cookie
                    //
                    Cookie cookies[] = req.getCookies();
                    String cookie = null;
                    if (cookies != null) {
                          for (int i = 0; i < cookies.length; i++) {
                                if (cookies[i].getName().equals("Session")) {
                                      cookie = cookies[i].getValue();
                                      break;
                                }
                          }
                    }
                    if (cookie != null && !mCookies.contains(cookie)) cookie =
null;
                    //
                    // show sign-in
                    //
                    if (cookie == null) {
                          res.setContentType("text/html");
                          PrintWriter pw = res.getWriter();
                          pw.println("<html><body><center>"
                                +"<h2>Sign-In</h2>"
                                +"<form method=post
action="+req.getServletPath()+">"
                                +"<input type=hidden name=Command value=SignIn>"
                                +"<table><tr><td>Username:</td>"
                                +"<td><input type=text name=Username
size=30></td>"
                                +"</tr><tr><td>Password:</td>"
                                +"<td><input type=password name=Password
size=30></td>"
                                +"</tr></table>"
                                +"<input type=submit value='OK'>"
                                +"</form>"
                                +"</center></body></html>");
                          return;
                    }
                    //
                    // do account
                    //
                    // get user hashtable which is used to keep all
information/values
                    // related to a particular user
                    // this approach is weak because the hashtable is not really
an
                    // object-oriented representation of the user and there is no
                    // type-checking for the values stored in the hashtable
                    Hashtable user =
(Hashtable)mUsers.get(cookie.substring(0,cookie.indexOf('.')));
                    String balance = (String)user.get("Balance");
                    if (cmd == null) {
                          showForm(req,res,balance);
                    } else if (cmd.equals("Deposit")) {
                          String value = req.getParameter("Value");
                          balance =
String.valueOf(Integer.parseInt(balance)+Integer.parseInt(value));
```


FIG. 6C

```
                        user.put("Balance",balance);
                        showForm(req,res,balance);
                    } else if (cmd.equals("Withdraw")) {
                        String value = req.getParameter("Value");
                        balance = String.valueOf(Integer.parseInt(balance)-
Integer.parseInt(value));
                        user.put("Balance",balance);
                        showForm(req,res,balance);
                    } else {
                        showForm(req,res,balance);
                    }
                } catch (Exception e) {
                    e.printStackTrace();
                }
    }
        private static final void showForm(HttpServletRequest
req,HttpServletResponse res,
                String balance) throws Exception {
                res.setContentType("text/html");
                PrintWriter pw = res.getWriter();
                pw.println("<html><body><center>"
                        +"<h2>Account</h2>"
                        +"<table>"
                        +"<tr><td>Balance:</td>"
                        +"<td>"+balance+"</td>"
                        +"<td> </td></tr>"
                        +"<form method=get action="+req.getServletPath()+">"
                        +"<input type=hidden name=Command value=Deposit>"
                        +"<tr><td>Deposit:</td>"
                        +"<td><input type=text name=Value size=30></td>"
                        +"<td><input type=submit value='OK'></td></tr>"
                        +"</form>"
                        +"<form method=get action="+req.getServletPath()+">"
                        +"<input type=hidden name=Command value=Withdraw>"
                        +"<tr><td>Withdraw:</td>"
                        +"<td><input type=text name=Value size=30></td>"
                        +"<td><input type=submit value='OK'></td></tr>"
                        +"</form>"
                        +"</table>"
                        +"</center></body></html>");
        }
    private static final String getRandomId() {
      // value range from 0 to 2147483648 inclusive
      Random r = new Random((new Date()).getTime());
      int rint = r.nextInt();
      rint = (rint < 0) ? -1*rint : rint;
      return String.valueOf(rint);
    }
        private static final void showError(HttpServletRequest
req,HttpServletResponse res,
                String message) throws Exception {
                res.setContentType("text/html");
                PrintWriter pw = res.getWriter();
                pw.println("<html><body><center>"
                        +"<h2>Error</h2>"
                        +"<p>"+message+"</p>"
                        +"<form method=get action="+req.getServletPath()+">"
```


FIG. 6D

```
                    +"<input type=submit value='OK'>"
                    +"</form>"
                    +"</center></body></html>");
            }
        }
```


FIG. 6E

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;

public class InventionDemo extends HttpServlet {
      private int balance;
      // there is type-checking for application values such as balance

  public void service(HttpServletRequest req,HttpServletResponse res)
    throws ServletException, IOException {
            try {
                  //
                  // do not need to check for sign-in
                  //
                  // a central user database can be maintained for all
applications
                  //
                  // system administration can assign access control in a
flexible manner
                  //
                  // do not need to get cookie
                  //
                  // do not need to show sign-in
                  //
                  // do not need to get user hashtable or session object
                  // which is used to keep all information/values
                  // related to a particular user
                  //
                  // this object is an object-oriented representation of this
application
                  // and there is type-checking for the values stored such as
balance
                  //
                  // go straight to do account
                  //
                  String cmd = req.getParameter("Command");
                  if (cmd == null) {
                        showForm(req,res,balance);
                  } else if (cmd.equals("Deposit")) {
                        String value = req.getParameter("Value");
                        balance += Integer.parseInt(value);
                        showForm(req,res,balance);
                  } else if (cmd.equals("Withdraw")) {
                        String value = req.getParameter("Value");
                        balance -= Integer.parseInt(value);
                        showForm(req,res,balance);
                  } else {
                        showForm(req,res,balance);
                  }
            } catch (Exception e) {
                  e.printStackTrace();
            }
  }
      private static final void showForm(HttpServletRequest
req,HttpServletResponse res,
```


FIG. 7A

```
                int balance) throws Exception {
                res.setContentType("text/html");
                PrintWriter pw = res.getWriter();
                pw.println("<html><body><center>"
                      +"<h2>Account</h2>"
                      +"<table>"
                      +"<tr><td>Balance:</td>"
                      +"<td>"+balance+"</td>"
                      +"<td> </td></tr>"
                      +"<form method=get action="+req.getServletPath()+">"
                      +"<input type=hidden name=Command value=Deposit>"
                      +"<tr><td>Deposit:</td>"
                      +"<td><input type=text name=Value size=30></td>"
                      +"<td><input type=submit value='OK'></td></tr>"
                      +"</form>"
                      +"<form method=get action="+req.getServletPath()+">"
                      +"<input type=hidden name=Command value=Withdraw>"
                      +"<tr><td>Withdraw:</td>"
                      +"<td><input type=text name=Value size=30></td>"
                      +"<td><input type=submit value='OK'></td></tr>"
                      +"</form>"
                      +"</table>"
                      +"</center></body></html>");
          }
     }
```


FIG. 7B